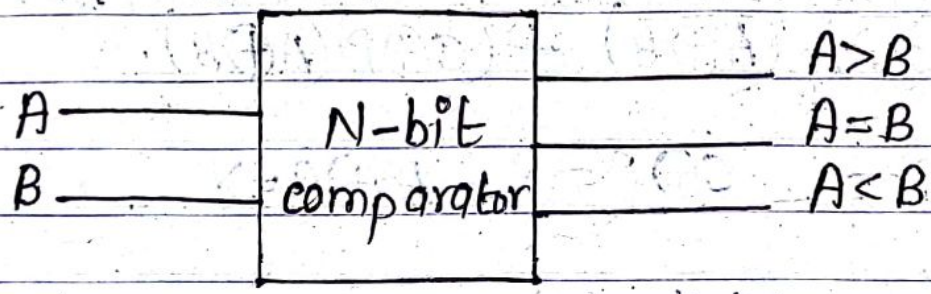
Magnitude Comparator:

A magnitude comparator is a combinational circuit that compares two binary numbers in order to find out whether one binary number is equal, less than or greater than the other binary number.

A magnitude comparator is a combinational circuit that compares two numbers, A and B, and determines their relative magnitudes. The outcome of the comparison is specified by three binary variables that indicate whether $A>B$, $A=B$, or $A<B$.



(a) 1-Bit magnitude comparator:

A comparator used to compare two bits is called a single bit comparator. It consists of two inputs each for two single bit numbers and three output to generate less than, equal to and greater than between two binary numbers.

The truth table for a 1-bit comparator is given below:

A	B	A<B	A=B	A>B
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

From the truth table, the logical expression for each output can be expressed as,

Using K-map, we get,

A<B

A \ B	B'	B
A'	0	1
A	0	0

A=B

A \ B	B'	B
A'	1	0
A	0	1

A>B

A \ B	B'	B
A'	0	0
A	1	0

$A<B = A'B$ $\quad A=B = A'B' + AB = A \odot B$ $\quad A>B = AB'$

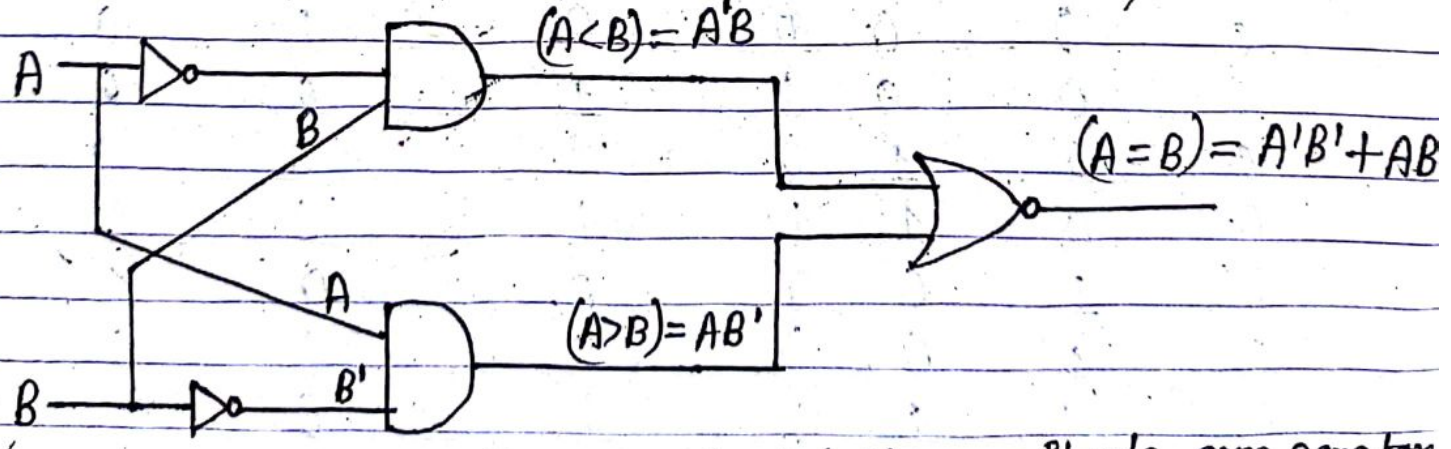


Fig: Logic diagram of 1-bit magnitude comparator

Since,

$$(A'B+AB')'$$
$$= (A'B)'(AB')'$$
$$= (A''+B')(A'+B'')$$
$$= (A+B')(A'+B)$$
$$= A(A'+B) + B'(A'+B)$$
$$= A\cdot A' + AB + A'B' + B'B$$
$$= AB + A'B'$$
$$= A \odot B$$

(b) 2-bit magnitude comparator:

A comparator used to compare two binary numbers each of two bits is called a 2-bit magnitude comparator. It consists of four inputs and three outputs to generate less than, equal to and greater than between two binary numbers.

The truth table of a 2 bit comparator is given below:

A_1	A_0	B_1	B_0	A<B	A=B	A>B
0	0	0	0	0	1	0
0	0	0	1	1	0	0
0	0	1	0	1	0	0
0	0	1	1	1	0	0
0	1	0	0	0	0	1
0	1	0	1	0	1	0
0	1	1	0	1	0	0
0	1	1	1	1	0	0

A_1	A_0	B_1	B_0	$A<B$	$A=B$	$A>B$
1	0	0	0	0	0	1
1	0	0	1	0	0	1
1	0	1	0	0	1	0
1	0	1	1	1	0	0
1	1	0	0	0	0	1
1	1	0	1	0	0	1
1	1	1	0	0	0	1
1	1	1	1	0	1	0

$A>B$

A_1A_0 \ B_1B_0	$B_1'B_0'$	$B_1'B_0$	B_1B_0	B_1B_0'
A_1A_0'	0	0	0	0
$A_1'A_0$	1	0	0	0
A_1A_0	1	1	0	1
A_1A_0'	1	1	0	0

$A<B$

A_1A_0 \ B_1B_0	$B_1'B_0'$	$B_1'B_0$	B_1B_0	B_1B_0'
$A_1'A_0'$	0	1	1	1
$A_1'A_0$	0	0	1	1
A_1A_0	0	0	0	0
A_1A_0'	0	0	1	0

$(A>B) = A_1B_1' + A_0B_1'B_0' + A_1A_0B_0'$

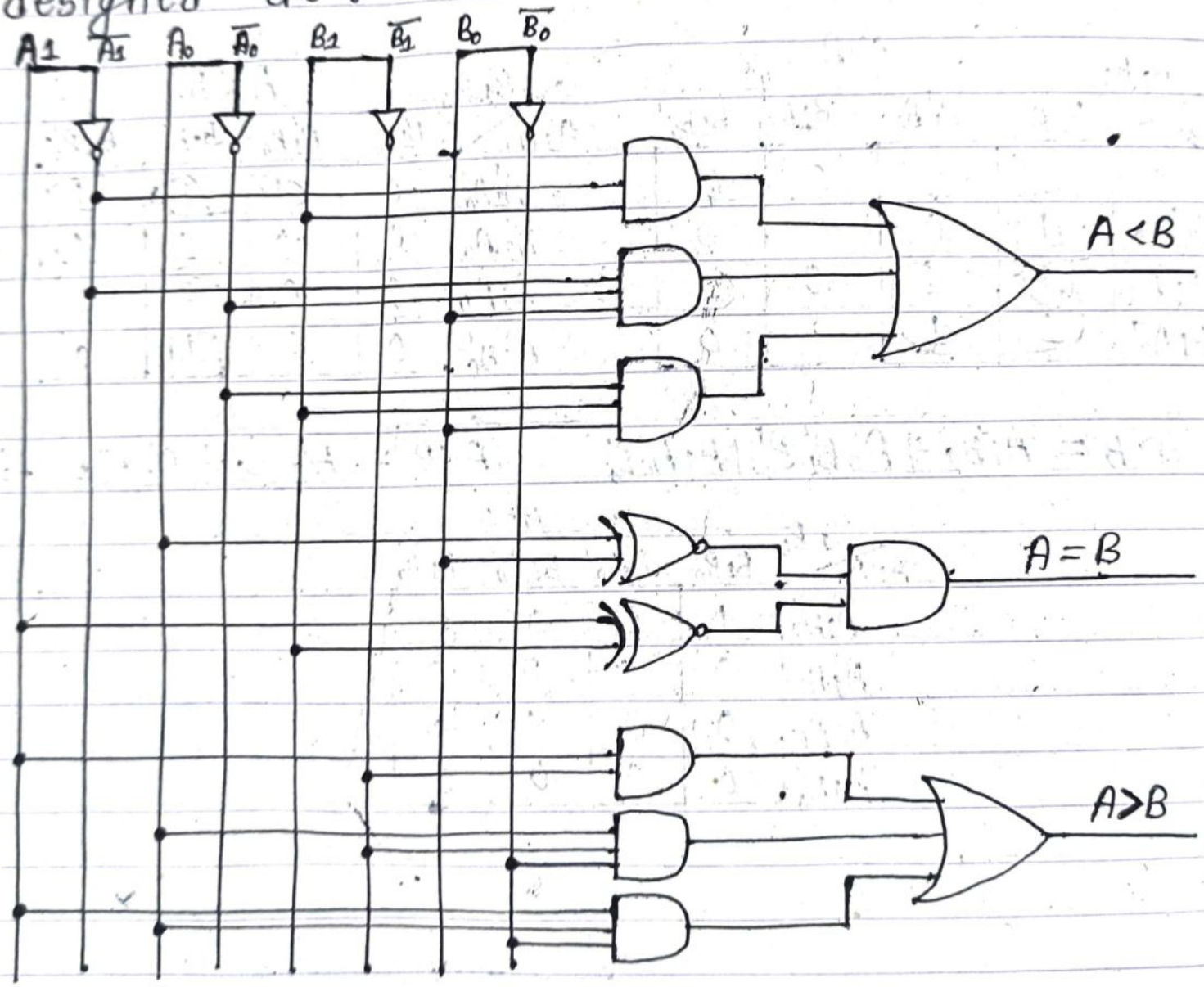
$(A<B) = A_1'B_1 + A_1'A_0'B_0 + A_0'B_1B$

$A=B$

A_1A_0 \ B_1B_0	$B_1'B_0'$	$B_1'B_0$	B_1B_0	B_1B_0'
$A_1'A_0'$	1	0	0	0
$A_1'A_0$	0	1	0	0
A_1A_0	0	0	1	0
A_1A_0'	0	0	0	1

$$(A=B) = A_1' A_0' B_1' B_0' + A_1' A_0 B_1' B_0 + A_1 A_0 B_1 B_0 + A_1 A_0' B_1 B_0'$$
$$= A_1' B_1' (A_0' B_0' + A_0 B_0) + A_1 B_1 (A_0 B_0 + A_0' B_0')$$
$$= (A_0 B_0 + A_0' B_0') (A_1' B_1' + A_1 B_1)$$
$$= (A_0 \odot B_0) \cdot (A_1 \odot B_1)$$
$$= (A_0 \text{ X-NOR } B_0) \cdot (A_1 \text{ X-NOR } B_1)$$

By using these Boolean expressions, a logic circuit for a 2-bit comparator can be designed as :-



(c) 4-bit magnitude comparator:

Consider two numbers, A and B, with four digits each. Write the coefficients of the numbers with descending significance as follows:

$A = A_3 A_2 A_1 A_0$

$B = B_3 B_2 B_1 B_0$

where each subscripted letter represents one of the digits in the number. The two numbers are equal if all pairs of significant digits are equal i.e. if $A_3 = B_3$, $A_2 = B_2$, $A_1 = B_1$ and $A_0 = B_0$

When the numbers are binary, the digits are either 1 or 0 and the equality relation of each pair of bits can be expressed logically with an equivalence function:

$$X_i = A_i B_i + A_i' B_i' \quad , i = 0, 1, 2, 3$$

where $X_i = 1$ only if the pair of bits in position i are equal, i.e. if both are 1's or both are 0's.

Algorithm:

(A = B)

For the equality condition to exist, all X_i variables must be equal to 1. This dictates an AND operation of all variables:

$(A=B) = x_3 x_2 x_1 x_0$

The binary variable $(A=B)$ is equal to 1 only if all pairs of the two numbers are equal.

$(A<B)$ or $(A>B)$

To determine if A is greater than or less than B, the relative magnitudes of pairs of significant digits are checked from the most significant position. If the two digits are equal, compare the next lower significant pair of digits. This comparison continues until a pair of unequal digits is reached.

$A>B$: If the corresponding digit of A is 1 and that of B is 0.

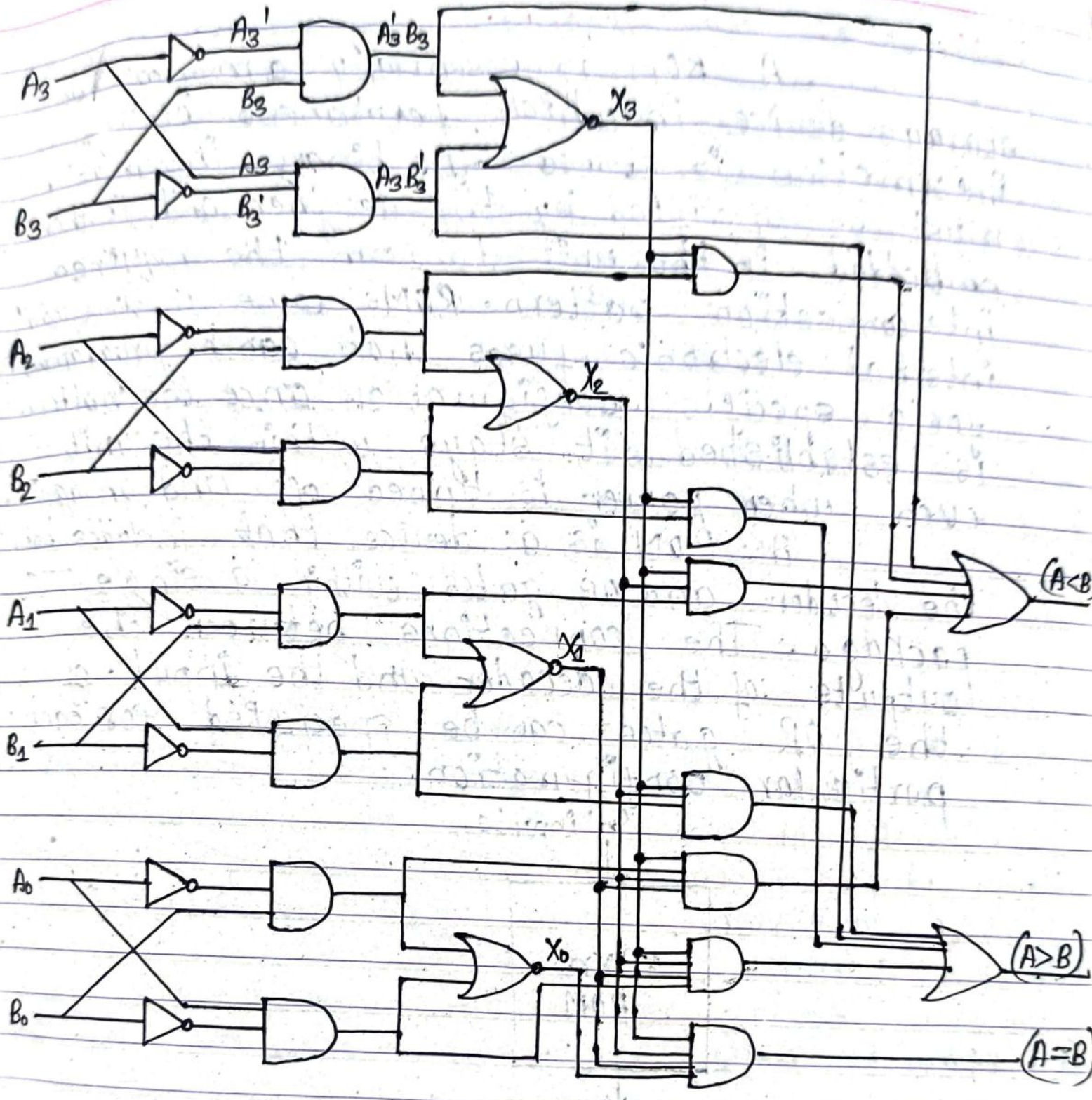
$A<B$: If the corresponding digit of A is 0 and that of B is 1.

The sequential comparison can be expressed logically by the following two Boolean functions.

$(A>B) = A_3 B_3' + x_3 A_2 B_2' + x_3 x_2 A_1 B_1' + x_3 x_2 x_1 A_0 B_0'$

$(A<B) = A_3' B_3 + x_3 A_2' B_2 + x_3 x_2 A_1' B_1 + x_3 x_2 x_1 A_0' B_0$

The logic diagram of 4-bit magnitude comparator is given as:-


A3
B3
A3'
B3'
A3'B3
A3B3'
X3
A2
B2
X2
A1
B1
X1
A0
B0
X0
(A<B)
(A>B)
(A=B)

ROM (Read Only Memory):

A ROM is essentially a memory or storage device in which permanent binary information is stored. The binary information must be specified by the designer and is then embedded in the unit to form the required interconnection pattern. ROMs come with special internal electronic fuses that can be programmed for a specific configuration once the pattern is established, it stays within the unit even when power is turned off and on again.

A ROM is a device that includes both the decoder and OR gates within a single IC package. The connections between the outputs of the decoder and the input of the OR gates can be specified for each particular configuration.

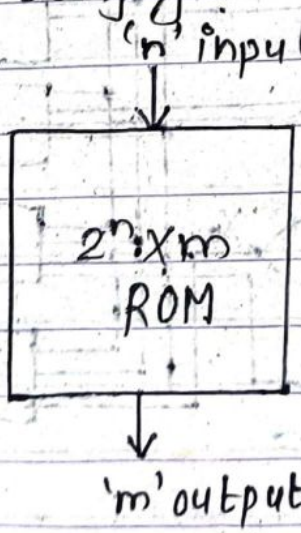
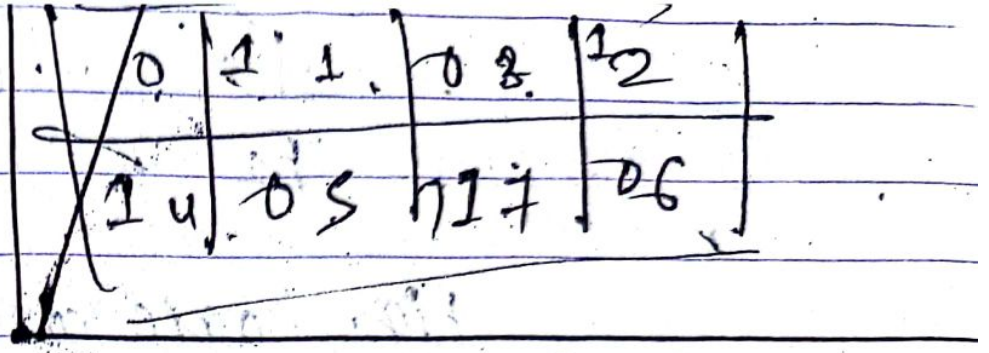


Fig: Block diagram of $2^n \times m$ ROM



A block diagram of ROM is shown in figure. It consists of n input lines and m output lines. Each bit combination of input variables is called an address. Each bit combination that comes out of the output lines is called a word. The number of bits per word is equal to the number of output lines 'm'. The number of distinct addresses possible with 'n' input variable is 2^n.

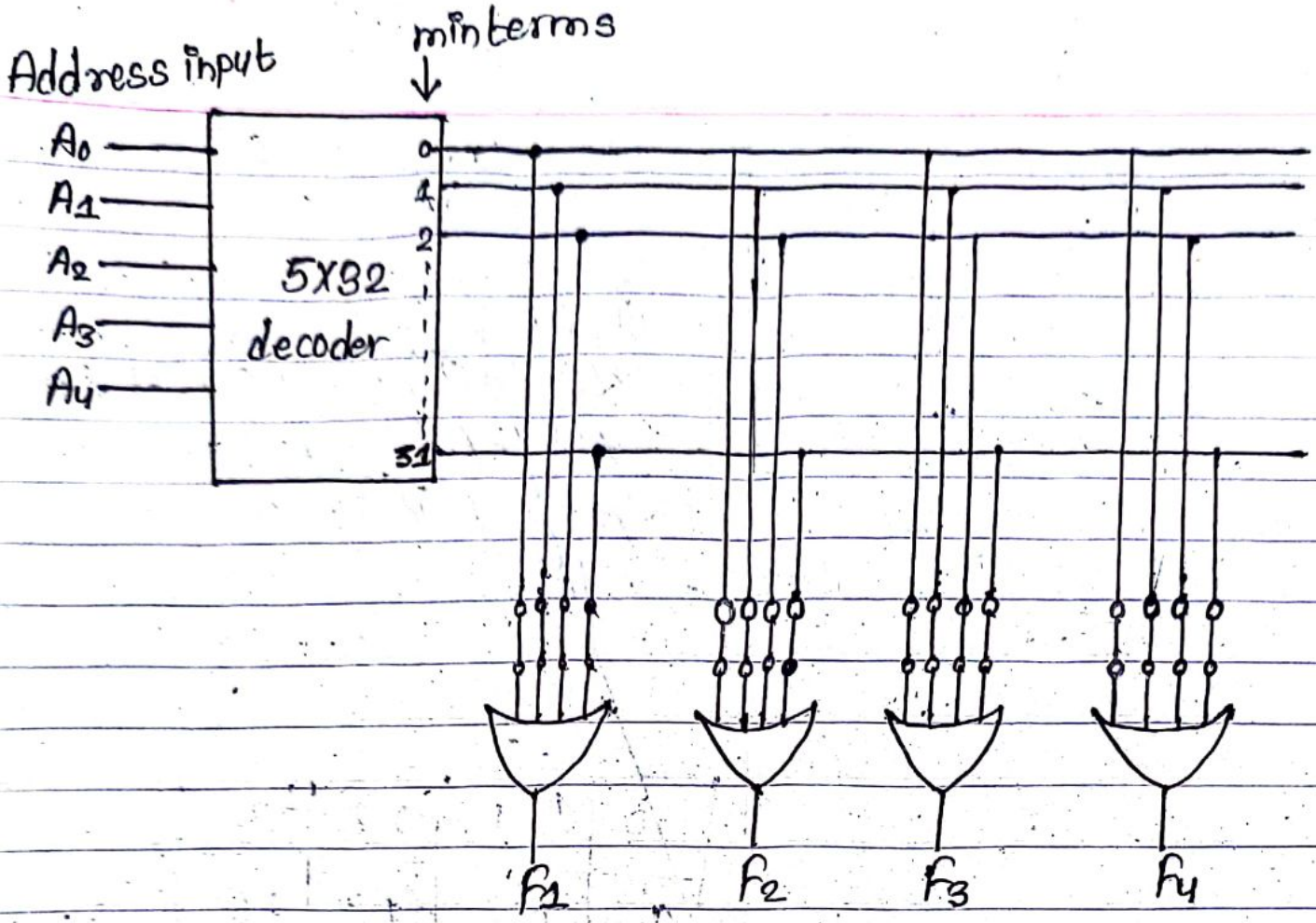


Fig: Logic construction of 32x5 ROM

The figure shows the logical construction of 32x5 ROM. It consists of 5 input variable that are decoded into 32 lines. Each output of the decoder represents one of the minter of a function of five variables. Each one of the 32 addresses selects one and only one output from the decoder. The addres is a 5-bit number applied to the inputs and the selected minterm out of the decoder is the one marked with the equivalent decimal number. The 32 outputs of the

decoder are connected through fuses (links) to each OR gate. only four of these links (fuses) are shown in the diagram, but actually each OR gate has 32 inputs and each input goes through a fuse (link) that can be broken as desired

(9) Design a combinational circuit using ROM.

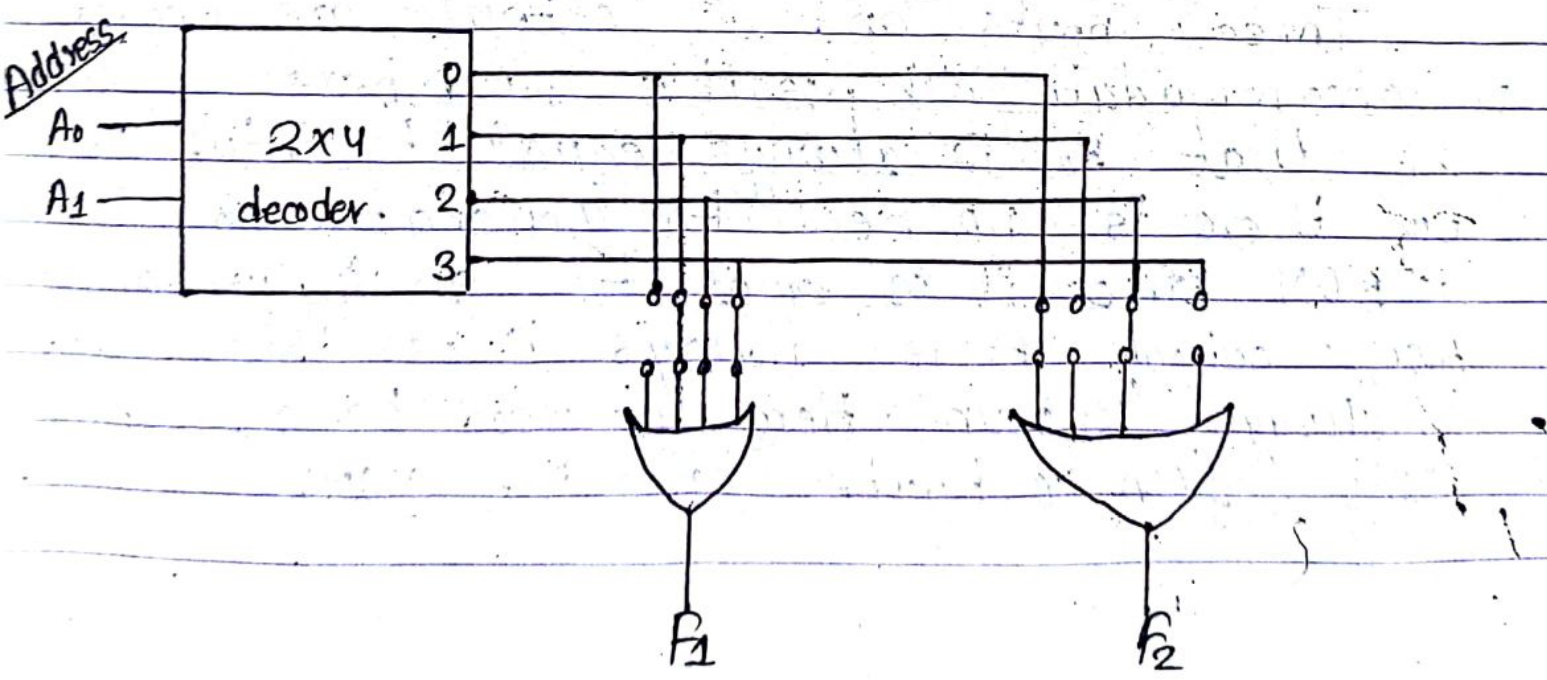
A_1	A_0	F_1	F_2
0	0	0	1
0	1	1	0
1	0	1	1
1	1	1	0

The Boolean function can be represented in SOP (minterm).

$F_1(A_1, A_0) = \Sigma(1,2,3)$

$F_2(A_1, A_0) = \Sigma(0,2)$

Since, there are two inputs, so,



(Q) Design a combinational circuit using a ROM. The circuit accepts a 3-bit number and generate an output binary number equal to the square of the input number.

⇒ The first step is to derive the truth table for the combinational circuit.

Inputs			Outputs						Decimal of
A_2	A_1	A_0	B_5	B_4	B_3	B_2	B_1	B_0	
0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	1	1
0	1	0	0	0	0	1	0	0	4
0	1	1	0	0	1	0	0	1	9
1	0	0	0	1	0	0	0	0	16
1	0	1	0	1	1	0	0	1	25
1	1	0	1	0	0	1	0	0	36
1	1	1	1	1	0	0	0	1	49

Three inputs and six outputs are needed to accommodate all possible numbers we can see that B_0 is always equal to input A_0. So, there is no need to generate B_0 with a ROM since it is equal to an input variable. Also, output B_1 is always 0; so this output is always known. Here, we need to generate only four outputs with the ROM.

The minimum size ROM needed must have three inputs and four outputs. Three input specify eight words, so, the ROM size must be 8x4.

The truth table given below specifies all the information needed for programming the ROM.

A_2	A_1	A_0	F_3	F_2	F_1	F_0
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	1
0	1	1	0	0	1	0
1	0	0	0	1	0	0
1	0	1	0	1	1	0
1	1	0	1	0	0	1
1	1	1	1	1	0	0

Fig: ROM truth table

The block diagram of required ROM is :-

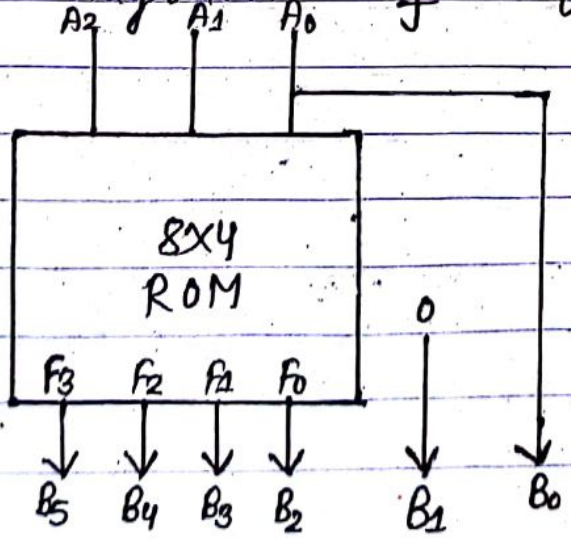


Fig: Block diagram of ROM

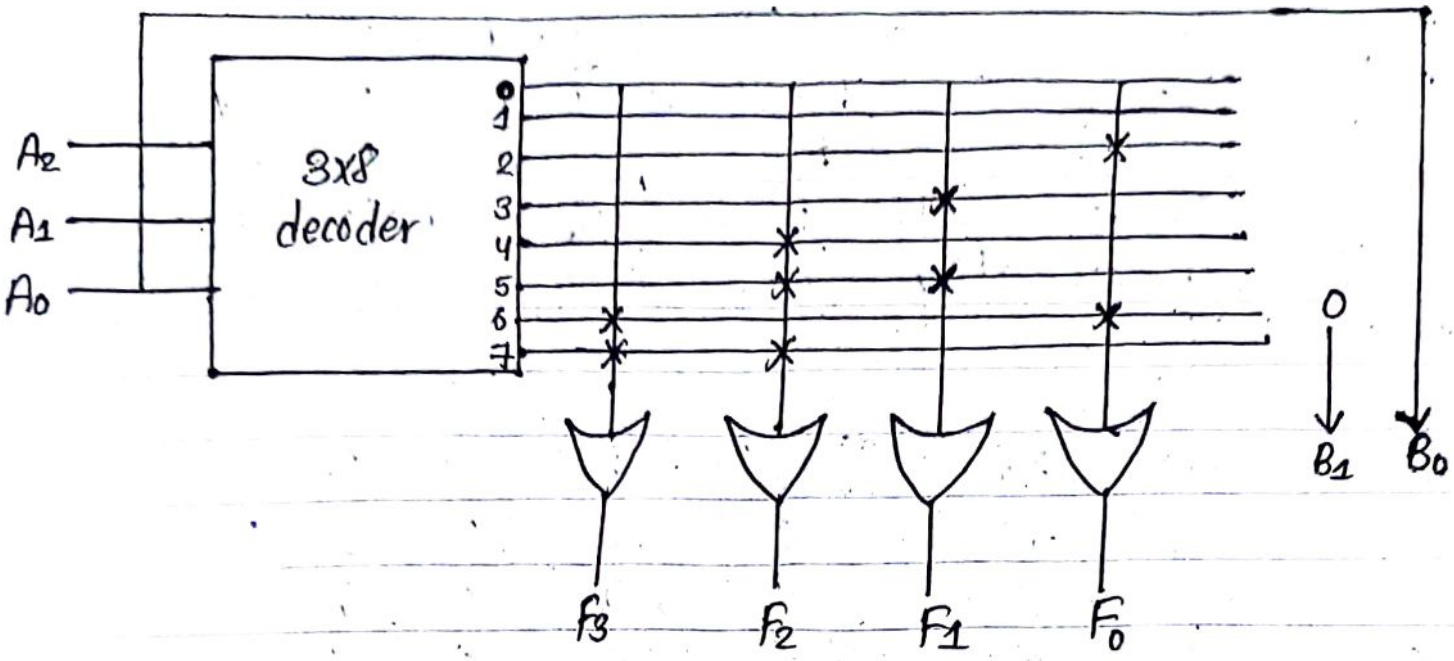


Fig: Logic diagram of ROM

20,21

[redacted] Jina